Parallel and Distributed Computing Assignment

Due: 23:59 on 18th Oct.

Problem I

Write a CUDA program to solve 2D-convlution in Deep Neural Network.

To make this problem easier, problem I only requires correctness of your program with following limitations, which means the performance of your program is not required in this problem.

Limitations:

1. The shape of input feature map should be [8, 64, 128, 128].
2. The shape of kernel should be [128, 64, 3, 3]
3. The data type of inputs is float.
4. The stride is 1.
5. There is no padding.
6. There is no bias.

If you are not familiar with convolution, following materials may help you.

Background of Convolution

Deep neural networks have become one of the most successful direction in the field of AI research. The most frequent operation of deep neural networks is the convolution operations. So, let's take a look at how the convolution

operation in the neural network is performed.

Here we only focus on 2D convolution. Convolution calculation includes two inputs: feature map and convolution kernel. The feature map is a 4-dimensional tensor of [N, C, H, W]. We can regard this 4-dimensional tensor as consisting of N sets of feature maps, each set of feature maps containing C channels, and each channel is an H*W 2D picture. The convolution kernel is a 4-dimensional tensor of [F, C, K, K]. Similarly, we can think of this 4-dimensional tensor as composed of F groups of convolution kernels, each group of convolution kernels contains C channels, and each channel is a K*K 2D matrix.

There is another parameter in the convolution calculation process: stride. It determines the size of the output result of the convolution calculation. But for the convenience of this homework, stride will be fixed to 1. Therefore, the size of the output Tensor is [N, F, H-K+1, W-K+1], we denote it as [N, F, H_, W_].

After clarifying all the inputs, outputs and parameters of the convolution calculation, the calculation process can be represented by the following python code:

```
for n in range(N):
    for f in range(F):
        for c in range(C):
            for h in range(H_):
                for w in range(W_):
                    for i in range(K):
                        for j in range(K):
                            output[n, f, h, w] += \
                            fm[n, c, h+i, w+j] * kernel[f, c, i, j]
```

In order to make it easier for you to understand, we present these two dynamic pictures from the Internet to explain the process vividly: Pircture I & Picture II.

Problem II

Optimize the program you just finished in problem I. Compare the performance of your optimized program, your original program and the CPU version of convolution program with the same input.

[Hints]

1. The limitations are not changed. So, you can make special optimizations for such input conditions.
2. Perhaps 2D convolution can be somehow calculated by matrix multiplication?
3. According to Hint.2, matrix multiplication can be easily optimized using shared memory then.
4. The CPU version code should be a single-thread, 7-loop code as in the background in problem I.

[Something Helpful]

- im2col
- shared memory
- nvcc

Requirement

Your submission should be able to run on server provided below successfully. And **notice that** your score will be extremely low if you use codes which can be found on Google or Baidu by searching "CUDA 卷积" because those codes cannot solve this problem and can be find out easily!

You can refer to the open source code on the Internet, but you must quote it correctly, and you must comment and explain the steps in the source code, or you will still get an extremely low score!

Environment

[Server IP] 202.120.38.28

[Port] 22

[Username] publichw2

[Password] cs427

Notice that everyone will use the **same** username & password pair, so you should make your own folder with your full name, e.g. "/home/publichw2/zhangsan" for San Zhang. And you are recommended to **back up** your files in case that someone else may remove your files by accident :)

Submission

There are **two** things you need to submit:

- Source Code
- Report

Your report should include optimization methods, comparison results and analysis of the results. Source code and report should be archived with name like "StudentID_Name_HW2.tar.gz" (or any other archive file types). Binary file is not needed.

If you have any questions, please feel free to contact TAs (吴飞洋、王雅洁、官惠泽) in our WeChat group.